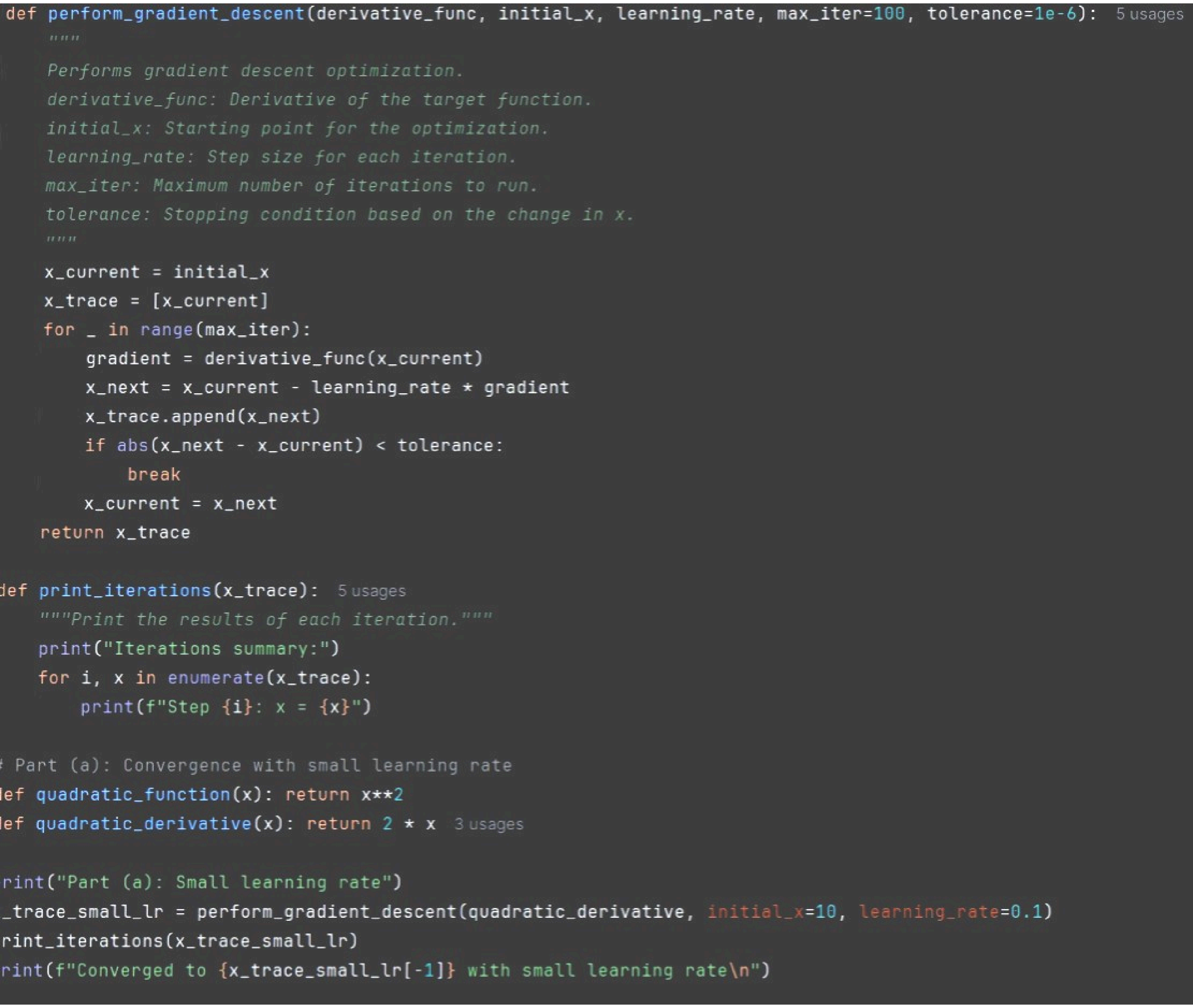
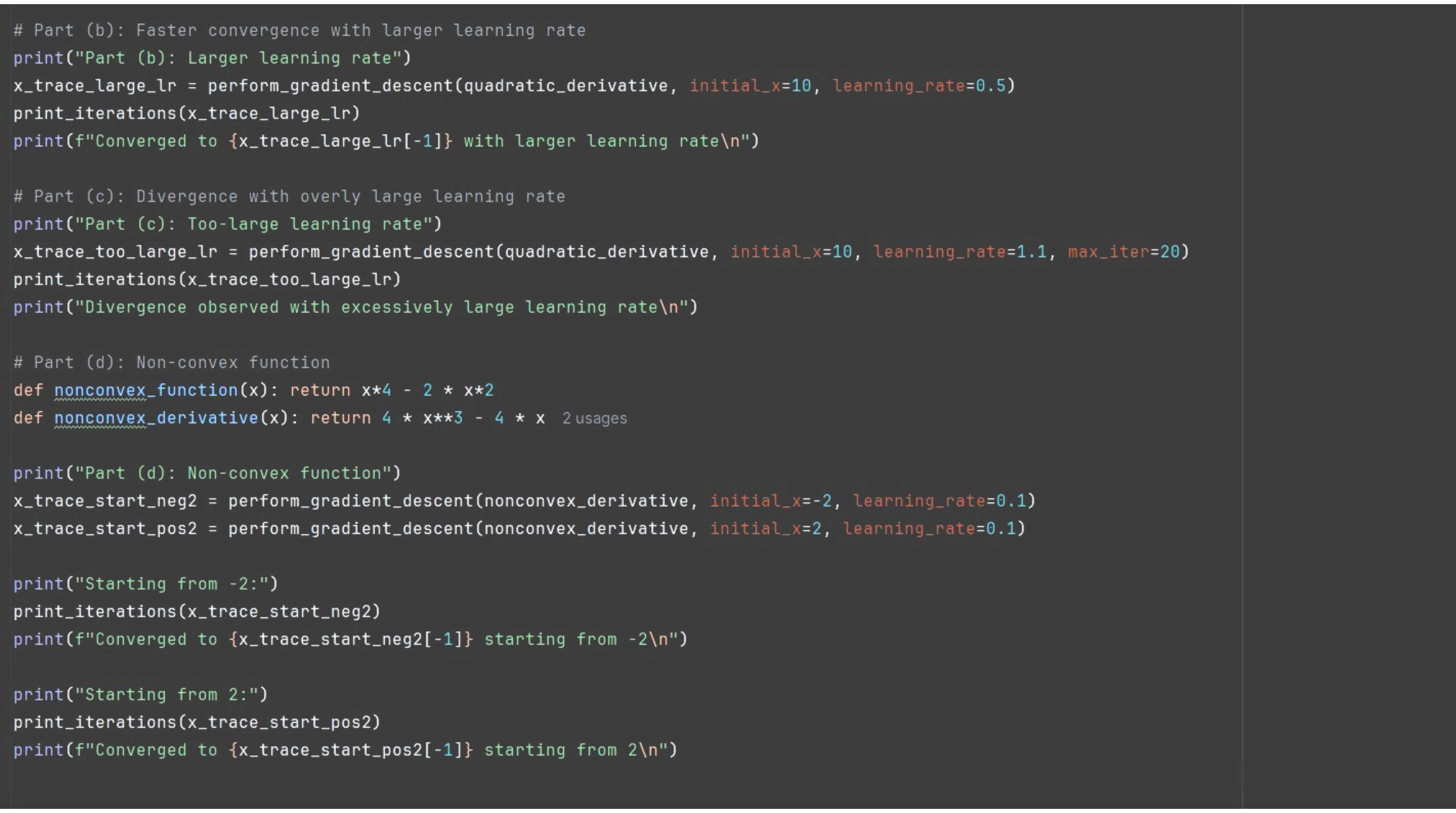
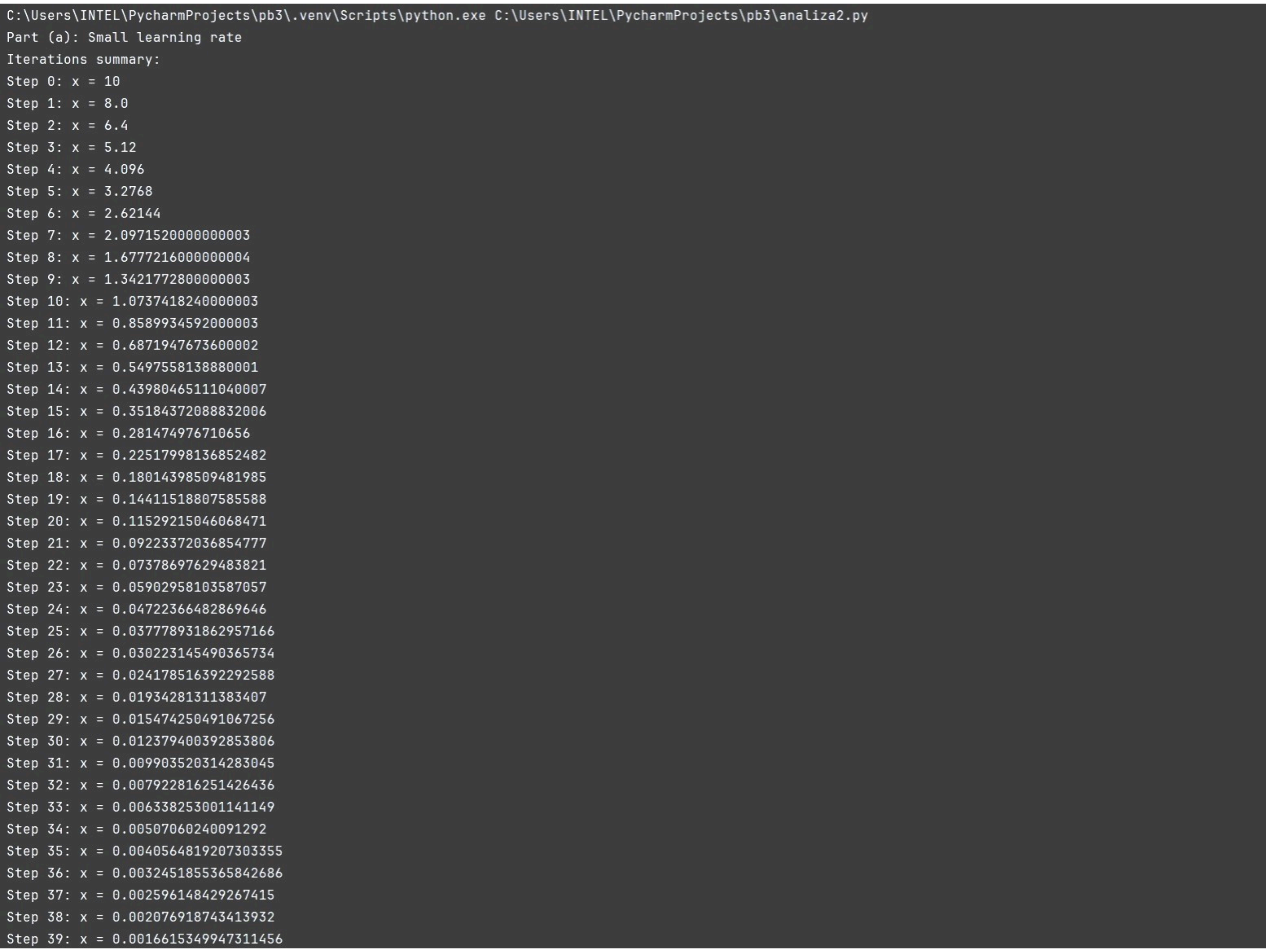


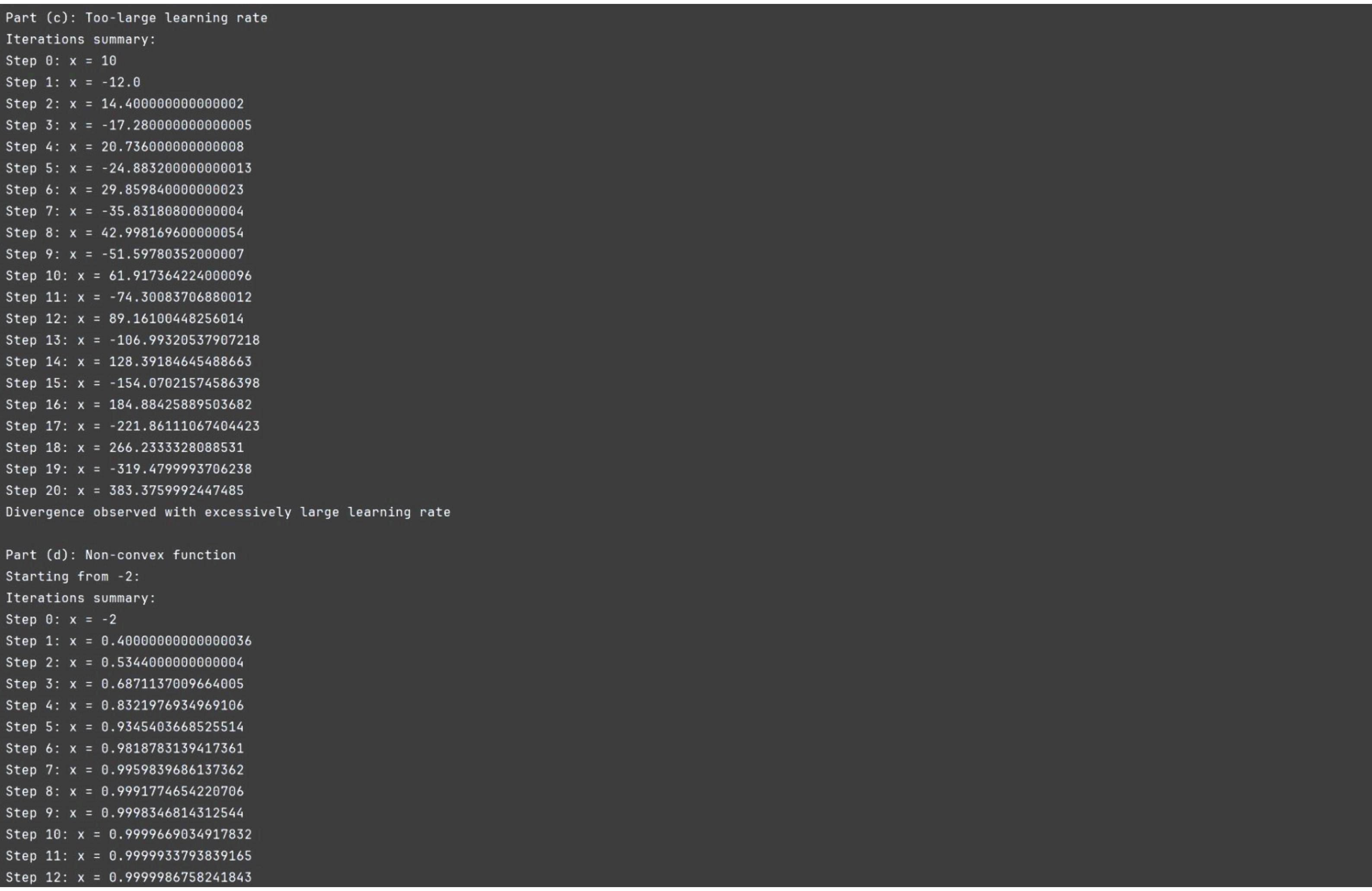
```
def perform_gradient_descent(derivative_func, initial_x, learning_rate, max_iter=100, tolerance=1e-6):  5 usages
    """
    Performs gradient descent optimization.
    derivative_func: Derivative of the target function.
    initial_x: Starting point for the optimization.
    learning_rate: Step size for each iteration.
    max_iter: Maximum number of iterations to run.
    tolerance: Stopping condition based on the change in x.
    """
    x_current = initial_x
    x_trace = [x_current]
    for _ in range(max_iter):
        gradient = derivative_func(x_current)
        x_next = x_current - learning_rate * gradient
        x_trace.append(x_next)
        if abs(x_next - x_current) < tolerance:
            break
        x_current = x_next
    return x_trace

def print_iterations(x_trace):  5 usages
    """Print the results of each iteration."""
    print("Iterations summary:")
    for i, x in enumerate(x_trace):
        print(f"Step {i}: x = {x}")

# Part (a): Convergence with small learning rate
lef quadratic_function(x): return x**2
lef quadratic_derivative(x): return 2 * x  3 usages

rint("Part (a): Small learning rate")
_trace_small_lr = perform_gradient_descent(quadratic_derivative, initial_x=10, learning_rate=0.1)
rint_iterations(x_trace_small_lr)
rint(f"Converged to {x_trace_small_lr[-1]} with small learning rate\n")
```



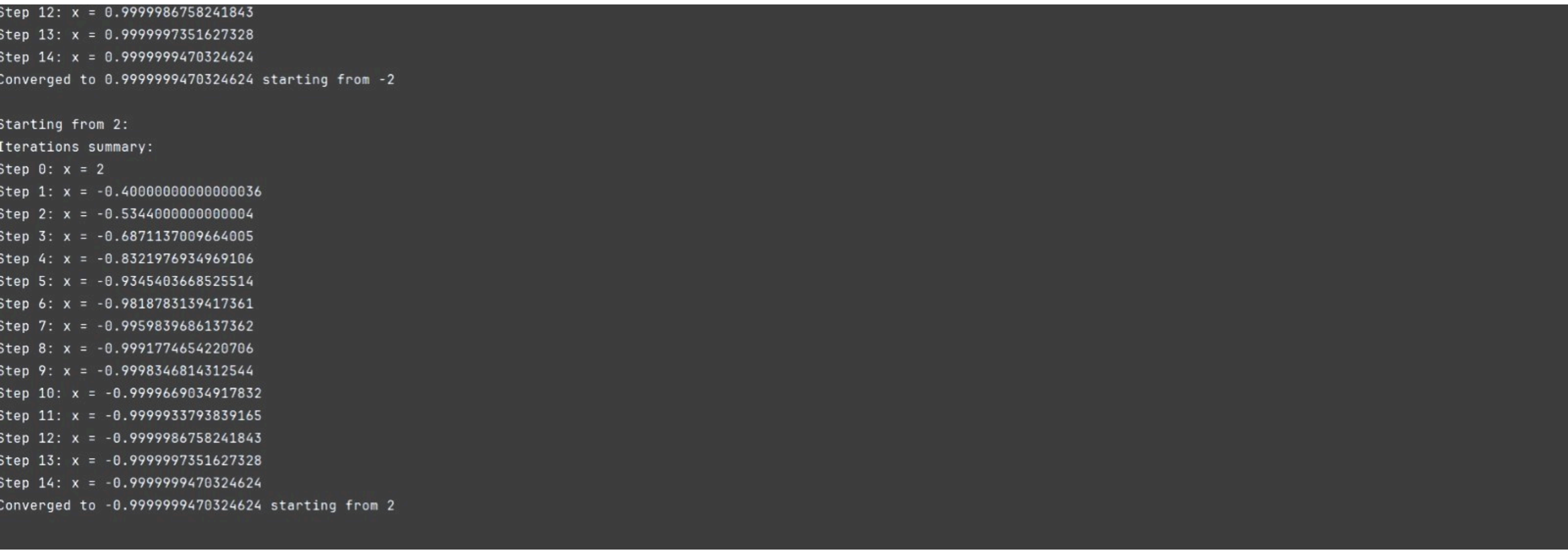
```
# Part (b): Faster convergence with larger learning rate
print("Part (b): Larger learning rate")
x_trace_large_lr = perform_gradient_descent(quadratic_derivative, initial_x=10, learning_rate=0.5)
print_iterations(x_trace_large_lr)
print(f"Converged to {x_trace_large_lr[-1]} with larger learning rate\n")

# Part (c): Divergence with overly large learning rate
print("Part (c): Too-large learning rate")
x_trace_too_large_lr = perform_gradient_descent(quadratic_derivative, initial_x=10, learning_rate=1.1, max_iter=20)
print_iterations(x_trace_too_large_lr)
print("Divergence observed with excessively large learning rate\n")

# Part (d): Non-convex function
def nonconvex_function(x): return x*4 - 2 * x*2
def nonconvex_derivative(x): return 4 * x**3 - 4 * x  2 usages

print("Part (d): Non-convex function")
x_trace_start_neg2 = perform_gradient_descent(nonconvex_derivative, initial_x=-2, learning_rate=0.1)
x_trace_start_pos2 = perform_gradient_descent(nonconvex_derivative, initial_x=2, learning_rate=0.1)

print("Starting from -2:")
print_iterations(x_trace_start_neg2)
print(f"Converged to {x_trace_start_neg2[-1]} starting from -2\n")

print("Starting from 2:")
print_iterations(x_trace_start_pos2)
print(f"Converged to {x_trace_start_pos2[-1]} starting from 2\n")
```



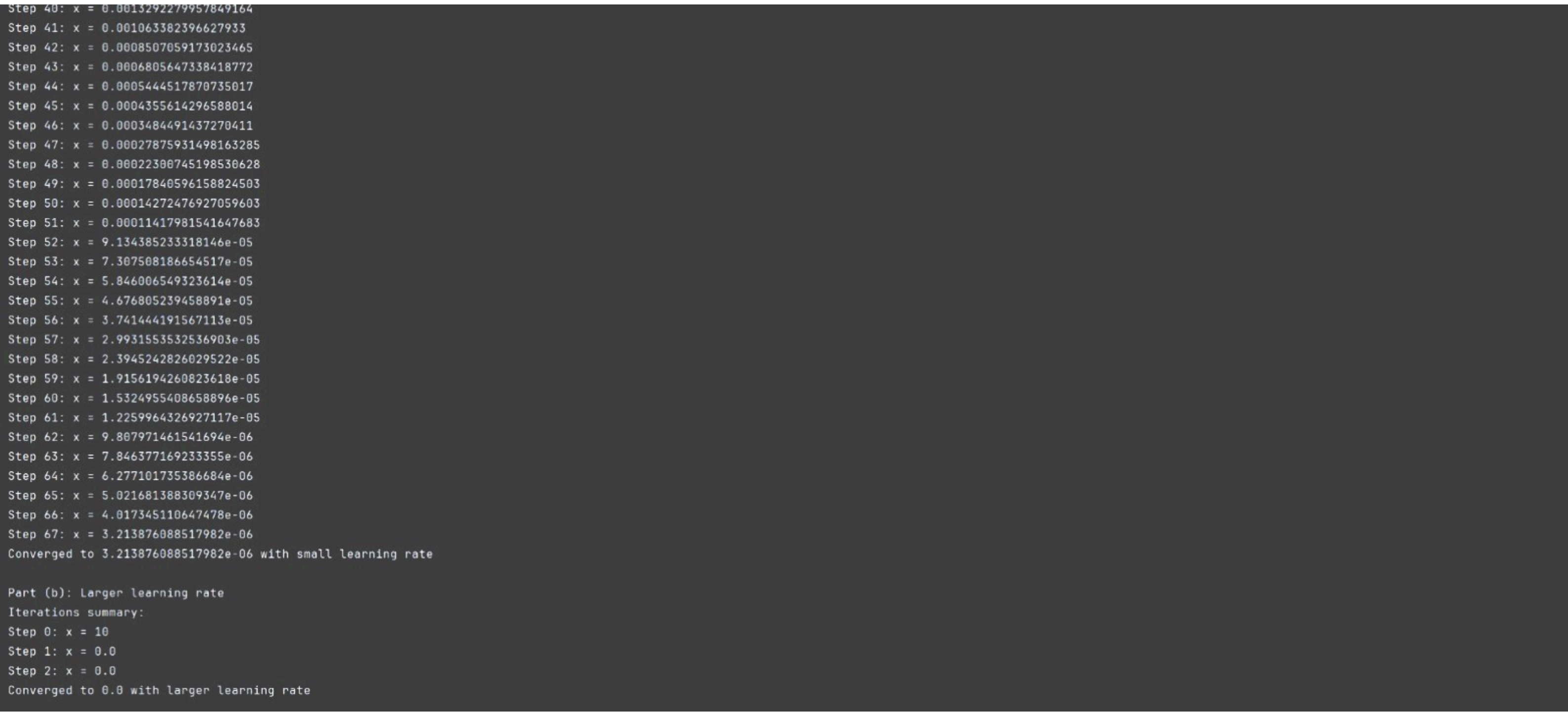
```
C:\Users\INTEL\PycharmProjects\pb3\.venv\Scripts\python.exe C:\Users\INTEL\PycharmProjects\pb3\analiza2.py
Part (a): Small learning rate
Iterations summary:
Step 0: x = 10
Step 1: x = 8.0
Step 2: x = 6.4
Step 3: x = 5.12
Step 4: x = 4.096
Step 5: x = 3.2768
Step 6: x = 2.62144
Step 7: x = 2.0971520000000003
Step 8: x = 1.6777216000000004
Step 9: x = 1.3421772800000003
Step 10: x = 1.0737418240000003
Step 11: x = 0.8589934592000003
Step 12: x = 0.6871947673600002
Step 13: x = 0.5497558138880001
Step 14: x = 0.43980465111040007
Step 15: x = 0.35184372088832006
Step 16: x = 0.281474976710656
Step 17: x = 0.22517998136852482
Step 18: x = 0.18014398509481985
Step 19: x = 0.14411518807585588
Step 20: x = 0.11529215046068471
Step 21: x = 0.09223372036854777
Step 22: x = 0.07378697629483821
Step 23: x = 0.05902958103587057
Step 24: x = 0.04722366482869646
Step 25: x = 0.037778931862957166
Step 26: x = 0.030223145490365734
Step 27: x = 0.024178516392292588
Step 28: x = 0.01934281311383407
Step 29: x = 0.015474250491067256
Step 30: x = 0.012379400392853806
Step 31: x = 0.009903520314283045
Step 32: x = 0.007922816251426436
Step 33: x = 0.006338253001141149
Step 34: x = 0.00507060240091292
Step 35: x = 0.0040564819207303355
Step 36: x = 0.0032451855365842686
Step 37: x = 0.002596148429267415
Step 38: x = 0.002076918743413932
Step 39: x = 0.0016615349947311456
```



```
Part (c): Too-large learning rate
Iterations summary:
Step 0: x = 10
Step 1: x = -12.0
Step 2: x = 14.400000000000002
Step 3: x = -17.280000000000005
Step 4: x = 20.736000000000008
Step 5: x = -24.883200000000013
Step 6: x = 29.859840000000023
Step 7: x = -35.83180800000004
Step 8: x = 42.998169600000054
Step 9: x = -51.59780352000007
Step 10: x = 61.917364224000096
Step 11: x = -74.30083706880012
Step 12: x = 89.16100448256014
Step 13: x = -106.99320537907218
Step 14: x = 128.39184645488663
Step 15: x = -154.07021574586398
Step 16: x = 184.88425889503682
Step 17: x = -221.86111067404423
Step 18: x = 266.2333328088531
Step 19: x = -319.4799993706238
Step 20: x = 383.3759992447485
Divergence observed with excessively large learning rate

Part (d): Non-convex function
Starting from -2:
Iterations summary:
Step 0: x = -2
Step 1: x = 0.40000000000000036
Step 2: x = 0.5344000000000004
Step 3: x = 0.6871137009664005
Step 4: x = 0.8321976934969106
Step 5: x = 0.9345403668525514
Step 6: x = 0.9818783139417361
Step 7: x = 0.9959839686137362
Step 8: x = 0.9991774654220706
Step 9: x = 0.9998346814312544
Step 10: x = 0.9999669034917832
Step 11: x = 0.9999933793839165
Step 12: x = 0.9999986758241843
```



```
Step 12: x = 0.9999986758241843
Step 13: x = 0.9999997351627328
Step 14: x = 0.9999999470324624
Converged to 0.9999999470324624 starting from -2

Starting from 2:
Iterations summary:
Step 0: x = 2
Step 1: x = -0.40000000000000036
Step 2: x = -0.5344000000000004
Step 3: x = -0.6871137009664005
Step 4: x = -0.8321976934969106
Step 5: x = -0.9345403668525514
Step 6: x = -0.9818783139417361
Step 7: x = -0.9959839686137362
Step 8: x = -0.9991774654220706
Step 9: x = -0.9998346814312544
Step 10: x = -0.9999669034917832
Step 11: x = -0.9999933793839165
Step 12: x = -0.9999986758241843
Step 13: x = -0.9999997351627328
Step 14: x = -0.9999999470324624
Converged to -0.9999999470324624 starting from 2
```



```
Step 40: x = 0.0013292279957849164
Step 41: x = 0.001063382396627933
Step 42: x = 0.0008507059173023465
Step 43: x = 0.0006805647338418772
Step 44: x = 0.0005444517870735017
Step 45: x = 0.0004355614296588014
Step 46: x = 0.0003484491437270411
Step 47: x = 0.00027875931498163285
Step 48: x = 0.00022300745198530628
Step 49: x = 0.00017840596158824503
Step 50: x = 0.00014272476927059603
Step 51: x = 0.00011417981541647683
Step 52: x = 9.134385233318146e-05
Step 53: x = 7.307508186654517e-05
Step 54: x = 5.846006549323614e-05
Step 55: x = 4.676805239458891e-05
Step 56: x = 3.741444191567113e-05
Step 57: x = 2.9931553532536903e-05
Step 58: x = 2.3945242826029522e-05
Step 59: x = 1.9156194260823618e-05
Step 60: x = 1.5324955408658896e-05
Step 61: x = 1.2259964326927117e-05
Step 62: x = 9.807971461541694e-06
Step 63: x = 7.846377169233355e-06
Step 64: x = 6.277101735386684e-06
Step 65: x = 5.021681388309347e-06
Step 66: x = 4.017345110647478e-06
Step 67: x = 3.213876088517982e-06
Converged to 3.213876088517982e-06 with small learning rate

Part (b): Larger learning rate
Iterations summary:
Step 0: x = 10
Step 1: x = 0.0
Step 2: x = 0.0
Converged to 0.0 with larger learning rate
```